## REMARKS

Prior to addressing each ground of rejection, applicants summarize here the environment that motivated applicants to invent the claimed microprocessor and method. Content providers may store various computer programs and data on a server, e.g., a server that maintains a web site that is associated with the content provider. They may want to restrict access to those programs and data to authorized computer systems. One way to restrict access is to limit distribution to computer systems that include registered processor numbers. Prior to delivering programs and/or data that a computer system requests, a server that is associated with the content provider may retrieve that system's processor number. The server may then compare that processor number with a registered set of processor numbers for computer systems that are authorized to access the programs and/or data. If the computer system's processor number matches one included in the registered set, then access is granted. Otherwise, access is denied.

Such a mechanism for restricting access may, however, enable one to track a user's Internet activity, which may raise user privacy concerns. To protect user privacy, the following mechanism has been proposed. Initially, a computer system requests access to a content provider's programs and/or data, which may be stored on a server that maintains a web site. In response, the server delivers to the computer system a key that is unique for the web site that stores the requested content, and requests the computer system to identify itself. The computer system does so by generating a hash value from its processor number and that key, then delivering the hash value to the server. The server then compares the hash value that the computer system returned with an expected hash value that is stored on the server. If they match, access is granted. If they do not match, access is denied.

Unfortunately, this mechanism for ensuring user privacy may be circumvented if multiple web sites, associated with various content providers, use the same key. For that reason, to prevent a user's Internet activity from being tracked, a computer system may need to confirm that a key corresponds to an intended web site prior to delivering to the web site a hash value that is derived from the key and the system's processor number. The difficulty here, as applicants explain in the "Background of the Invention" section of the specification (at page 4, lines 3-10), is that the mechanism for checking whether a key corresponds to a web site may enable an unauthorized user to access content stored on the web site.

Accordingly, prior to applicants' invention, there existed a need for an improved device and method that ensured restricted access of programs and/or data to authorized computer systems while preserving user privacy. In response to that need, applicants devised the microprocessor of pending claim 1, which includes: (1) an identifier that identifies the microprocessor, and (2) embedded instructions for comparing a hash value, derived from the identifier and a key, to an expected hash value. The claimed microprocessor enables a server to authenticate a computer system, which includes the claimed microprocessor, by sending a key and an expected hash value to the computer system, then waiting for a response that is representative of that system's identity. The microprocessor facilitates that response by generating a hash value from the key and the identifier, then comparing that hash value with the expected hash value that the server provided. In a significant departure from prior authentication processes, the claimed microprocessor – not the server – performs the hash value comparison operation, then provides the result to the server.

Applicants' novel microprocessor facilitates the method of applicants' claim 12 and the computer-readable medium of applicants' claim 20. In that method, an application (e.g., one from a web site stored on a server) transmits a key and an expected hash to a computer system. A hash value is generated from the key and a second identifier (e.g., a processor number for the computer system's microprocessor). That hash value is then compared with the expected hash value that the application transmitted. The computer-readable medium of claim 20 includes computer-executable instructions that may facilitate the claimed method.

Authentication processes that preceded the method of pending claim 12 require the computer system to return a hash value to the server. The server then compares the hash value with an expected hash value that is stored on the server. Unlike those previous methods, applicants' claimed method requires the server to deliver the expected hash value to the computer system. The computer system's microprocessor then performs the hash value comparison operation instead of the server. By enabling that operation to be shifted from the server to the computer system, applicants' claimed microprocessor and method provide a hardware based mechanism for authenticating a computer system. Such a hardware dependent approach may provide a more reliable way to perform such an authentication function, while maintaining user privacy, when compared to the software based processes that preceded applicants' claimed invention.

**Rejection Under 35 U.S.C. §102 Based on Mi et al.**

The examiner rejected claims 1-8, 10, 12, 17, and 20 under 35 U.S.C. §102(e) as being anticipated by Mi et al. Applicants have canceled claims 7, 8, 10, and 17, obviating the need to respond to this rejection as it pertains to those claims. Mi does not anticipate the microprocessor of claims 1-6, the method of claim 12, or the computer-readable medium of claim 20.

The microprocessor of claim 1 comprises (1) an identifier that identifies the microprocessor, and (2) embedded instructions for comparing a hash value, derived from the identifier and a key, to an expected hash value. Although Mi describes a microprocessor that includes an identifier that identifies the microprocessor, Mi does not describe a microprocessor that includes embedded instructions for comparing a hash value, derived from the identifier and a key, to an expected hash value.

The process Mi describes for performing a hash value comparison operation is like those identified in the "Background of the Invention" section of the pending application. That process includes the following steps. A computer system requests access to an object that is stored on a server. In response, the server delivers to the computer system certain programming instructions that are included in a “verification agent” or a DLL file. Those instructions, when executed by the computer system’s microprocessor, retrieve that microprocessor’s processor number, generate a hash value derived in part from the processor number, and return the hash value to the server. The server then compares that hash value with an expected hash value that is stored on the server.

The process that Mi describes for authenticating a computer system, which requests access to an object stored on a server, does not employ instructions that are embedded within a microprocessor to perform a hash value comparison operation. Nor does Mi describe a microprocessor that includes embedded instructions for performing that operation. For that reason, Mi does not anticipate the microprocessor of pending claim 1.

Nor does Mi anticipate the microprocessor of claim 2. Because claim 2 depends upon claim 1, Mi fails to anticipate the microprocessor of claim 2 for that reason alone. In addition, Mi does not anticipate that claim because Mi does not describe a microprocessor that includes embedded instructions for producing a hash value that is a function of the identifier and a key. Mi does not employ instructions that are embedded within a microprocessor to produce such a hash value. Instead, Mi describes generating a hash value derived in part from a processor number and a key by executing programming instructions that are included in a “verification agent” or DLL file that is downloaded from a server. Mi does not describe a microprocessor that includes embedded instructions for performing that operation. For that additional reason, Mi does not anticipate the microprocessor of pending claim 2.

Claim 4 specifies that the embedded instructions comprise microcode. Because Mi does not describe a microprocessor with embedded instructions for performing either a hash value generation function or a hash value comparison function, it must follow that Mi does not describe a microprocessor that comprises microcode for performing such functions. To support a contrary view, the examiner improperly relies on Mi's reference (at column 3, lines 47-50) to storing embedded information for identifying a processor in a constant ROM that holds data that microcode instructions may use. That reference does not indicate that the microprocessor Mi describes includes microcode for performing hash value generation and comparison operations. Rather, it simply references U.S. Patents 5,790,834 and 5,794,066, which describe a microprocessor that includes microcode for accessing the microprocessor's processor number. Enclosed with this amendment are copies of those two patents, which confirm that the excerpt from Mi, upon which the examiner relies, describes nothing more than microcode instructions for enabling an application to access a processor number.

Because Mi does not describe a microprocessor that includes embedded instructions that comprise microcode for performing hash value generation and comparison operations, Mi does not anticipate the microprocessor of pending claim 4 for this additional reason.

In addition to failing to anticipate the microprocessor of pending claims 1-6, Mi does not anticipate the method of claim 12 or the computer-readable medium of claim 20. The method of claim 12 requires that an application (e.g., one from a web site stored on a server) transmit a request to a computer system to confirm the identity of the computer system. That request is "accompanied by a key and an expected hash value derived from that key and a first identifier for a computer system." That expected hash value is then compared to a hash value that is derived from a second identifier and the key. The computer-readable medium of claim 20 stores instructions for carrying out a method like the method of claim 12.

As explained previously, Mi's process transmits instructions from a server to a computer system that (1) retrieve a processor number; (2) generate a hash value derived in part from the processor number; and (3) return the hash value to the server. The instructions that Mi's process transmits to the computer system from the server are not "accompanied by a key and an <u>expected hash value</u> derived from that key and a first identifier for a computer system." In Mi's process, the expected hash value remains stored on the server. Unlike the process of applicants' claim 12, Mi's process does not deliver an expected hash value to a computer system to enable the computer system -- not the server -- to compare that expected hash value to a hash value derived in part from the computer system's processor number.

Because Mi does not describe a process that transmits "a request from an application to a computer system to confirm the identity of the computer system, the request accompanied by a key <u>and an expected hash value</u> derived from that key and a first identifier for a computer system," Mi does not anticipate the method of claim 12, or the computer-readable medium of claim 20. Because Mi does not describe a processor, process, or computer-readable medium that anticipates the processor, process, or computer-readable medium of applicants' claims 1-6, 12, and 20, applicants respectfully request the examiner to withdraw the rejection of those claims based on that reference.

**Rejection Under 35 U.S.C. §102 Based on Granger et al.**

The examiner rejected claims 1-4, 7-9, 12-14, 16-18, and 20 under 35 U.S.C. §102(e) as being anticipated by Granger et al. Applicants have canceled claims 7-9 and 17-18, obviating the need to respond to this rejection as it pertained to those claims. Granger does not anticipate the microprocessor of claims 1-4, the methods of claims 12-14 and 16, or the computer-readable medium of claim 20.

Granger describes a system for copy protection that relies upon an Electronic Security Device ("ESD"), which may be implemented as a unit of a computer microprocessor. A software developer may program the ESD with a "secret value" before the ESD is shipped. To protect a software application using such a pre-programmed ESD, the application delivers a seed value to the ESD, which calculates a response value. That response value is then compared with an expected response value. If the values match, the application is executed. If not, then the application will close.

In the specific example that Granger describes, a 64-bit seed value is delivered to an ESD. The ESD applies a one-way hash function to the seed value and a 64-bit K-value, which the software developer programmed into the ESD prior to shipping it. The ESD then returns the resulting 64-bit response value. That 64-bit response value may be used as a key for encrypting user data or decrypting encrypted user date, as shown in Granger's figures 1A and 1B.

Granger does not anticipate the microprocessor of claim 1, which comprises (1) an identifier that identifies the microprocessor, and (2) embedded instructions for comparing a hash value, derived from the identifier and a key, to an expected hash value. The identifier of claim 1 must vary from one microprocessor to the next. Otherwise, one could not rely upon the identifier to distinguish one microprocessor from another.

Granger does not anticipate a microprocessor that comprises "an identifier that identifies the microprocessor." Granger, unlike applicants, was not concerned with computer system authentication, which requires each system to possess a unique identifier. Instead, Granger addressed ways to prevent unauthorized copying of computer programs. Granger's system pre-programs an ESD with a "secret value" (e.g., a 64-bit "K-value") prior to shipping it to customers. That "secret value" is used to generate a return value (e.g., a "key") for decryption and encryption operations. Prior to allowing the return value to be used in such operations, the application first checks to ensure that it matches an expected return value.

Granger's system only works if the "secret value," which the software developer programs into the ESD and which "is maintained in secrecy by the software developer," is identical for every ESD that is shipped. Unless that "secret value" is identical for every ESD, a unique "expected return value" must be calculated for every ESD shipped. It is impractical, if not impossible, for an application to contain customized "secret values" for every ESD shipped. If different "secret values" were programmed into different ESDs, then the application would have to be updated every time a new ESD was shipped to add to it a new "secret value" and a new "expected return value" that is derived from the "secret value." No one would consider Granger to describe a system that required applications to be constantly updated every time a new ESD was shipped.

For these reasons, it will be evident to anyone skilled in the art that Granger must describe a system that programs an identical "secret value" into every ESD. In Granger's system, an application may use such a universally applied "secret value" to generate return values for comparison to expected return values, which the application calculates from the pre-programmed universal "secret value" – which is "maintained in secrecy by the software developer" – and various seed values. Because Granger's pre-programmed "secret value" is not "an identifier that identifies a microprocessor," Granger fails to anticipate the microprocessor of claim 1 for that reason alone.

Granger does not anticipate the microprocessor of applicants' claim 1 for a second reason. The claimed microprocessor includes embedded instructions for comparing a hash value, derived from the identifier and a key, to an expected hash value. Granger does not describe a microprocessor that includes such embedded instructions. Like the system Mi describes, Granger's system performs a hash value comparison operation that is similar to those identified in the "Background of the Invention" section of the pending application. Granger's hash value comparison operation serves to validate a return value (e.g., a "key") that is used to encrypt data or decrypt encrypted data.

Like Mi, the system Granger describes for verifying a return value does not employ instructions that are embedded within a microprocessor to perform a hash value comparison operation. On the contrary, Granger indicates that "the application compares the response value to the expected response value." (See Granger at column 5, lines 14-16.) Because Granger's system uses the application – not the ESD – to compare response values, Granger does not describe employing instructions that are embedded within a microprocessor to perform a hash value comparison operation.

Because Granger does not describe a microprocessor that includes embedded instructions for performing a hash value comparison operation, Granger fails to anticipate the microprocessor of claim 1 for this additional reason. Because claims 2-4 depend upon claim 1, it must follow that Granger does not anticipate those claims either.

Moreover, Granger does not anticipate the microprocessor of claim 4 for an additional reason. Claim 4 specifies that the embedded instructions comprise microcode. Because Granger does not describe a microprocessor with embedded instructions for performing a hash value comparison function, it must follow that Granger does not describe a microprocessor that comprises microcode for performing that function. The excerpt from Granger, upon which the examiner relies when concluding that this reference identifies a microprocessor that includes microcode for comparing a hash value to an expected hash value, does not indicate that the ESD includes microcode for performing that operation. On the contrary, that excerpt clearly indicates that the application (which includes expected response values that the software developer generated during the development of the application) – not the ESD – performs the return value comparison.

Because Granger does not describe a microprocessor that includes embedded instructions that comprise microcode for performing a hash value comparison operation, Granger does not anticipate the microprocessor of claim 4 for this additional reason.

In addition to failing to anticipate the microprocessor of pending claims 1-4, Granger does not anticipate the method of claim 12 or the computer-readable medium of claim 20. Claim 12 specifies "a method for confirming the identity of a computer system." That claim requires an application (e.g., one from a web site stored on a server) to transmit a request to a computer system "to confirm the identity of the computer system." That request is accompanied by a key and an expected hash value derived from that key and "a first identifier for a computer system." The computer-readable medium of claim 20 stores instructions for carrying out a method like the method of claim 12.

Granger does not describe "a method for confirming the identity of a computer system." Granger's method relates instead to preventing unauthorized copying of computer programs. To prevent unauthorized copying, Granger's method pre-programs an ESD with a "secret value," which enables an application to check whether computer systems can execute the application without subjecting it to unauthorized copying. The identity of individual computer systems is of no concern to Granger. Granger's method only seeks to ensure that otherwise anonymous computer systems include a widely distributed ESD prior to allowing those systems to execute an application, which is restricted to authorized users only.

Because Granger does not describe "a method for confirming the identity of a computer system," Granger does not anticipate the method of claim 12 for that reason alone. In addition, Granger does not describe a method that transmits a request to a computer system that is accompanied by a key and an expected hash value derived from that key and "a first identifier for a computer system." The expected response value that Granger's application uses may be derived from a seed value and a "64-bit K-value," which a software developer may program into the ESDs. As explained previously, such a "K-value" cannot differ from one ESD to another, as such a practice would require the application to be updated every time a new ESD was shipped.

Because such a "K-value" is not "a first identifier for a computer system," Granger does not describe a method that transmits a request to a computer system that is accompanied by an expected hash value derived from "a first identifier for a computer system." For that additional reason, Granger does not anticipate the method of claim 12, or the computer-readable medium of claim 20.

Because claims 13, 14 and 16 depend upon claim 12, it must follow that Granger does not anticipate those claims either. Moreover, claim 16 requires that a "true response" be returned to the decryption program if processor numbers match, and a "false response" be returned it they do not match. As explained in applicants' specification, returning a "true" or "false" response -- after the computer system performs the hash value comparison operation -- ensures that neither the computer system's processor number nor a hash value derived from that number is exposed when an application performs an identity check on the computer system. Delivering such a reply in response to an application's authentication request preserves user privacy in an optimum fashion.

Unlike the method of claim 16, Granger's process returns a hash value – not a "true" or "false" response – to the application, when the application checks whether a computer system is authorized to execute it. If the hash value matches an expected hash value, the computer system may execute the application. If the hash values do not match, execution is closed. Granger does not generate a "true response," which is returned to a decryption program, if processor numbers are identical, or a "false response" if those numbers are not identical – as claim 16 requires. Even if Granger's process could authenticate individual computer systems (it cannot, for the reasons presented above), the mechanism for doing so would expose a hash value to the application. That is precisely the event that the claim 16 method prevents by returning "true" or "false" responses to an application that seeks to authenticate a computer system – such responses masking the hash value generated for the hash value comparison operation, and thereby preventing an application from tracking user activity.

The excerpt from Granger that the examiner contends teaches this process step (e.g., Granger at column 5, lines 10-25) does not do so. On the contrary, that excerpt simply describes comparing a response value to an expected response value, then permitting or closing continued execution depending upon the outcome of that comparison. That excerpt does not describe returning a true response if first and second processor numbers are identical, and returning a false response if first and second processor numbers are not identical. For that reason, Granger's process fails to anticipate the method of applicants' claim 16 for this additional reason.

Because Granger does not describe a processor, process, or computer-readable medium that anticipates the processor, process, or computer-readable medium of applicants' claims 1-4, 12-14, 16, and 20, applicants respectfully request the examiner to withdraw the rejection of those claims based on that reference.

**Rejection Under 35 U.S.C. §103 Based on Granger and Matsumoto**

The examiner rejected claims 10, 11 and 19 under 35 U.S.C. §103(a) as being unpatentable over Granger in view of Matsumoto. Applicants have canceled claims 10, 11 and 19, obviating the need to respond to this rejection.

**Rejection Under 35 U.S.C. §103 Based on Srinivasan and Granger**

The examiner rejected claim 21 under 35 U.S.C. §103(a) as being unpatentable over Srinivason in view of Granger. Applicants have canceled claim 21, obviating the need to respond to this rejection.

**Rejection Under 35 U.S.C. §103 Based on Granger and Calamera**

The examiner rejected claims 5, 6 and 15 under 35 U.S.C. §103(a) as being unpatentable over Granger in view of Calamera. Claims 5 and 6 depend, directly or indirectly, upon claim 4. Because claim 4 would not have been obvious in view of the combination of Granger and Calamera, it must follow that claims 5 and 6 would not have been obvious in view of that combination.

As explained in detail above, Granger does not describe a microprocessor that comprises "an identifier that identifies the microprocessor" or "embedded instructions for comparing a hash value, derived from the identifier and a key, to an expected hash value." In addition, Granger does not describe a microprocessor in which those embedded instructions comprise microcode, as claim 4 requires. Because Granger is concerned with copy protection, and not computer system authentication, Granger does not offer any suggestion that would have motivated one skilled in the art to modify a microprocessor so that it includes microcode for comparing hash values. On the contrary, because Granger teaches that the application – and not Granger's ESD – should perform any response value comparison (see Granger at column 5, lines 14-16), Granger effectively teaches away from adding microcode to a microprocessor to perform a hash value comparison operation. Calamera does not offer any teaching that compensates for Granger's deficiency.

Because neither reference describes a microprocessor with microcode for performing a hash value comparison operation, or provides any teaching that would have motivated one skilled in the art to modify a microprocessor to include microcode for performing that function, the combination of Granger and Calamera could not have rendered the microprocessor of claim 4 obvious. Because claims 5 and 6 depend upon claim 4, it must follow that those references' combination could not have rendered obvious the microprocessor of those claims either. For that reason, applicants respectfully request the examiner to withdraw the rejection of claims 5 and 6 based on the combination of Granger and Calamera.

Claim 15 depends, indirectly, upon claim 12. Because claim 12 would not have been obvious in view of the combination of Granger and Calamera, it must follow that claim 15 would not have been obvious in view of that combination. As explained in detail above, Granger does not describe "a method for confirming the identity of a computer system," or transmitting a request to a computer system that is accompanied by an expected hash value that is derived from "a first identifier for a computer system."

Because Granger is concerned with copy protection, and not computer system authentication, Granger does not offer any suggestion that would have motivated one skilled in the art to modify the process Granger describes to add steps for "confirming the identity of a computer system." Likewise, given that

Granger's process is directed toward copy protection – not computer system authentication, Granger does not offer any suggestion that would have motivated one skilled in the art to modify any process Granger describes to add steps for transmitting a request to a computer system that is accompanied by an expected hash value that is derived from "a first identifier for a computer system."

Calamera does not offer any teaching that compensates for Granger's deficiency. Because neither reference describes a method for "confirming the identity of a computer system" or for transmitting a request to a computer system that is accompanied by an expected hash value that is derived from "a first identifier for a computer system," and because neither reference provides any teaching that would have motivated one skilled in the art to modify any process they describe to include those process steps, the combination of Granger and Calamera could not have rendered the method of claim 12 obvious. Because claim 15 depends upon claim 12, it must follow that those references' combination could not have rendered obvious the method of that claim either. For that reason, applicants respectfully request the examiner to withdraw the rejection of claim 15 based on the combination of Granger and Calamera.

**Rejection Under 35 U.S.C. §103 Based on Srinivasan, Granger and Calamera**

The examiner rejected claims 22 and 23 under 35 U.S.C. §103(a) as being unpatentable over Srinivason and Granger further in view of Calamera. Applicants have canceled claims 22 and 23, obviating the need to respond to this rejection.

**Rejection Under 35 U.S.C. §103 Based on Srinivasan, Granger, Calamera and Matsumoto**

The examiner rejected claim 24 under 35 U.S.C. §103(a) as being unpatentable over Srinivason, Granger and Calamera further in view of Matsumoto. Applicants have canceled claim 24, obviating the need to respond to this rejection.

**Conclusion**

Because none of the cited references describe the processor, process, or computer-readable medium of applicants' claims 1-6, 12-16, and 20, or offer any teaching – either when considered alone or in combination – that would have rendered obvious that processor, process, or computer-readable medium, applicants respectfully request the examiner to withdraw the rejections of those claims based on the cited references, and to allow them to issue.

Respectfully submitted,

Date: March 25, 2004

Mark Seeley

Mark Seeley
Reg. No: 32,299
ATTORNEY FOR APPLICANT

Intel Corporation
Mail Stop SC4-202
2200 Mission College Blvd.
Santa Clara, CA 95052-8119
(408) 765-7382